Examinations 1(3)

- confirmed examination dates
  1. November 24, 2003
  2. February 2, 2004
  3. March 29, 2004
- check the exact times and places at **http://www.it.utu.fi/opetus/tentit/**
- remember to enroll!

Examinations 2(3)

- if you are *not* a student of University of Turku, you must register to receive the credits
- further instructions are available at **http://www.tucs.fi/Education/Information/regcredits.php**

Examinations 3(3)

- questions
  - based on both lectures and lecture notes
  - four questions, à 8 points
  - to pass the examination, at least 16 points (50%) are required
  - questions are in English, but you can answer in English or in Finnish

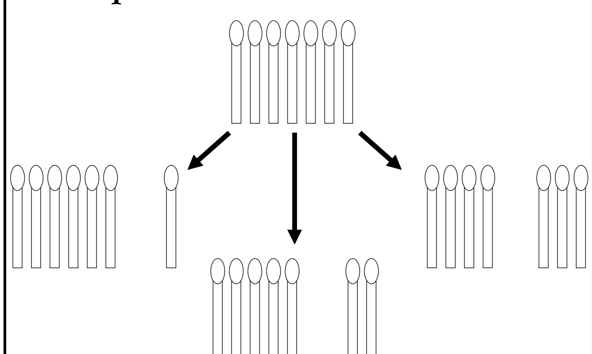
§3 Game Trees

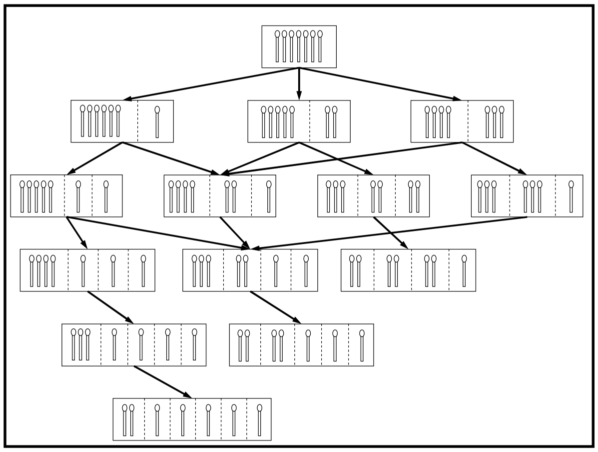
- perfect information games
  - no hidden information
- two-player, perfect information games
  - Noughts and Crosses
  - Chess
  - Go
- imperfect information games
  - Poker
  - Backgammon
  - Monopoly

Game tree

- all possible plays of two-player, perfect information games can be represented with a game tree
  - nodes: positions (or states)
  - edges: moves
- players: MAX (has the first move) and MIN
- ply = the number of edges from the root
  - MAX has even plies
  - MIN has odd plies

Simple Nim with seven matches





Problem statement

Given a node v in a game tree

find a winning strategy for MAX (or MIN) from v

or (equivalently)

show that MAX (or MIN) can force a win from v

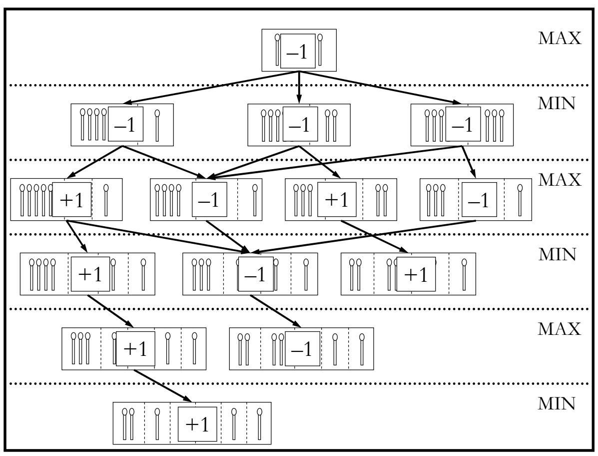
Minimax

- assumption: players are rational and try to win
- given a game tree, we know the outcome in the leaves
  - assign the leaves to win, draw, or loss (or a numeric value like +1, 0, –1) according to MAX's point of view
- at nodes one ply above the leaves, we choose the best outcome among the children (which are leaves)
  - MAX: win if possible; otherwise, draw if possible; else loss
  - MIN: loss if possible; otherwise, draw if possible; else win
- recurse through the nodes until in the root

Minimax rules

1. If the node is labelled to MAX, assign it to the maximum value of its children.
2. If the node is labelled to MIN, assign it to the minimum value of its children.

- MIN minimizes, MAX maximizes → minimax



Analysis (using negamax)

- simplifying assumptions
  - internal nodes have the same branching factor b
  - game tree is searched to a fixed depth d
- time consumption is proportional to the number of expanded nodes in negamax
  - 1 — root node (the initial ply)
  - b — nodes in the first ply
  - b^2 — nodes in the second ply
  - b^d — nodes in the dth ply
- overall running time $O(b^d)$

Rough estimates on running times when $d = 5$

- suppose expanding a node takes 1 ms
- branching factor b depends on the game
- Draughts ($b \approx 3$): $t = 0.243$ s
- Chess ($b \approx 30$): $t = 6\frac{3}{4}$ h
- Go ($b \approx 300$): $t = 77$ a
- alpha-beta pruning reduces b

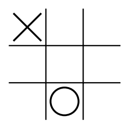
Choosing search depth

- usually the whole game tree is too large → limit search depth → a partial game tree
- n-move look-ahead strategy
  - stop searching after n moves
  - make the internal nodes leaves
  - use an evaluation function to 'guess' the outcome

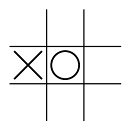
Example: Noughts and Crosses

- heuristic evaluation function e:
  - count the winning lines open to MAX
  - subtract the number of winning lines open to MIN
- forced wins
  - state is evaluated $+\infty$, if it is a forced win for MAX
  - state is evaluated $-\infty$, if it is forced win for MIN

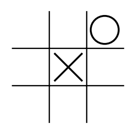
Examples of the evaluation



$e(n) = 6 - 5 = 1$



$e(n) = 4 - 6 = -2$



$e(n) = 5 - 4 = 1$

Drawbacks of the look-ahead approach

- horizon effect
  - heuristically promising path can lead to an unfavourable situation
  - solution: extend look-ahead on promising nodes → does not remove the problem
- bias
  - we want to have an estimate of minimax but get a minimax of estimates